ภาษา C

จเร เลิศสุดวิชัย

SOFT MAIL

พัฒนาโปรแกรมยูทิลิตี้ด้วย เทอร์โบซี

ตอนที่ 7

นำเสนอ 2 โปรแกรมที่ช่วยค้นหาข้อความในไฟล์ และช่วยเปลี่ยนตัวอักษรให้เป็นตัวเล็กหรือใหญ่ตามต้องการ

มุมภาษาซีฉบับนี้เสนอโปรแกรมยูทิลิตี้ช่วยงาน 2 โปรแกรมด้วยกัน ซึ่งเป็นโปรแกรมช่วยงานในด้านของไฟล์ข้อความ (Text File) โดยโปรแกรมแรกคือโปรแกรม FC (File-Case) ซึ่งทำหน้าที่เปลี่ยนตัวอักษรในไฟล์ให้เป็นตัวอักษรเล็กหรือใหญ่ตามที่ผู้ใช้ต้องการ และโปรแกรมที่สองคือ TS (Text-Search) เป็นโปรแกรมที่ทำงานคล้ายโปรแกรม Grep ซึ่งเป็นโปรแกรมช่วยที่นักเขียนโปรแกรมทั่วไปรู้จักดี เพราะโปรแกรมนี้จะช่วยค้นหาตำแหน่งของข้อความที่กำหนดในไฟล์ข้อความว่าอยู่ในบรรทัดใด เทคนิคและการทำงานของแต่ละโปรแกรมมีอย่างละเอียดดังนี้

โปรแกรม FC (File Case)

การทำงาน

โปรแกรมยูทิลิตี้นี้จะช่วยทำหน้าที่เปลี่ยนตัวอักษรของไฟล์ข้อความทั้งหมดให้เป็นตัวอักษรใหญ่ (Upper Case) หรือตัวอักษรเล็ก (Lower Case) ซึ่งผู้ใช้สามารถเลือกได้ตามต้องการโดยระบุไว้ที่คอมมานด์ไลน์

ผู้ใช้งานเครื่องไมโครคอมพิวเตอร์หลายคนจะพบว่า หลังจากการติดตั้งซอฟต์แวร์ใหม่ลงในเครื่องแล้ว ซอฟต์แวร์หลายตัวจะมีการเปลี่ยนแปลงสภาพแวดล้อมของระบบให้เหมาะสมกับการทำงานของซอฟต์แวร์เหล่านั้น ซึ่งการเปลี่ยนแปลงสภาพแวดล้อมนั้นจะเป็นการเพิ่มหรือเปลี่ยนข้อความในไฟล์ Autoexec.bat และไฟล์ Config.sys การเพิ่มหรือเปลี่ยนข้อความในไฟล์ทั้งสองนี้ในซอฟต์แวร์แต่ละโปรแกรมก็ไม่เหมือนกัน บางโปรแกรมก็ใช้ตัวอักษรตัวใหญ่ และบางโปรแกรมก็ใช้ตัวอักษรตัวเล็ก ซึ่งหากมีการติดตั้งซอฟต์แวร์หลายโปรแกรมในเครื่องก็จะมีผลทำให้ข้อความในไฟล์ทั้งสองมีทั้งตัวอักษรใหญ่และเล็กปนกันอยู่ ทำให้เกิดความไม่เรียบร้อยในไฟล์ทั้งสอง และโดยทั่วไปผู้ใช้งานมักจะชอบให้ข้อความในไฟล์ทั้งสองมีลักษณะเดียวกันคือ บางท่านอาจจะชอบอักษรตัวใหญ่ทั้งหมด ส่วนบางท่านชอบตัวอักษรเล็กทั้งหมด

ซึ่งในกรณีเช่นนี้โปรแกรมยูทิลิตี้ FC จะสามารถช่วยงานด้านนี้ได้เป็นอย่างดี เพราะโปรแกรม FC จะอ่านข้อความจากไฟล์ออกมาแล้วเปลี่ยนเป็นตัวอักษรใหญ่หรือเล็กตามต้องการ ซึ่งระบุเลือกได้ที่คอมมานด์ไลน์ โดยตัวอักษรที่เปลี่ยนแล้วจะถูกเขียนกลับลงไฟล์เดิม

จากการทำงานของโปรแกรม FC จะเห็นว่านอกจากจะสามารถนำไปใช้งานกับไฟล์ Autoexec.bat และไฟล์ Config.sys แล้ว โปรแกรม FC ยังสามารถใช้ไฟล์ข้อความใด ๆ ก็ได้ โดยสามารถทำงานได้อย่างถูกต้อง

การใช้งาน

```
fc filename.ext /c
```

filename.ext คือชื่อและนามสกุลของไฟล์ข้อความ ส่วน C นั้นคือตัวอักษร U หรือ L โดย U หมายถึงการระบุให้โปรแกรม fc เปลี่ยนตัวอักษรในไฟล์เป็นตัวอักษรใหญ่ (UPPER CASE) ส่วน L หมายถึงการเปลี่ยนเป็นตัวเล็ก การระบุชื่อไฟล์ผู้ใช้สามารถระบุพาธ (path) หรือไดเรคทอรีที่เป็นที่อยู่ของไฟล์ได้ด้วย

ตัวอย่าง

```
c: \ >fc autoexec.bat /u
```

ตัวอย่างนี้เป็นการเปลี่ยนตัวอักษรในไฟล์ autoexec.bat ให้เป็นตัวใหญ่

```
c: \ >fc config.sys /1
```

ตัวอย่างนี้เป็นการเปลี่ยนตัวอักษรในไฟล์ config.sys ให้เป็นตัวเล็ก

```
c: \ >fc autoexec.bat
```

ตัวอย่างนี้เป็นการเรียกโปรแกรม fc ทำงานโดยขาดการระบุชนิดของตัวอักษร โดยโปรแกรมจะไม่ทำงาน แต่จะรายงานข้อผิดพลาดพร้อมทั้งแสดงวิธีใช้โปรแกรมให้

```
c: \ >fc
```

หากเรียกโปรแกรมโดยไม่ระบุอะไรเลย โปรแกรมจะแสดงวิธีใช้

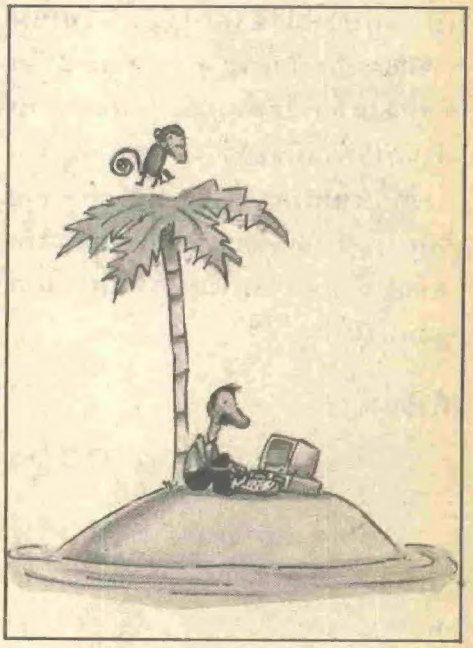
เทคนิคโปรแกรม

โปรแกรม fc เป็นตัวอย่างโปรแกรมที่แสดงถึงวิธีการตรวจสอบหาข้อเลือกหรือสวิตช์ที่กำหนดการทำงานของโปรแกรม โดยการทำงานของโปรแกรมจะเริ่มต้นจากฟังก์ชัน main โดยหลังจากนำพารามิเตอร์ตัวแรกไปใช้เป็นชื่อไฟล์แล้วก็จะทำการเชื่อมพารามิเตอร์ตัวอื่นในคอมมานด์ไลน์ที่เหลือเข้าด้วยกัน เก็บไว้ตัวแปรชื่อ Options ซึ่งจะได้รับการเปลี่ยนเป็นอักษรตัวใหญ่ก่อน จากนั้นก็จะตรวจสอบหาสวิตช์ /U และ /L โดยใช้ฟังก์ชัน strstr (Option, "/U") และฟังก์ชัน strstr (Options, "/L") การทำงานของฟังก์ชันจะตรวจสอบข้อความในพารามิเตอร์ตัวแรกว่ามีข้อความในพารามิเตอร์ตัวที่ 2 ปรากฏอยู่หรือไม่ ซึ่งในโปรแกรมก็คือการหาข้อความ "/U" และ "/L" ในตัวแปร Options นั่นเอง (ฟังก์ชัน strstr จะถูกใช้งานอีกครั้งในโปรแกรม Ts-Text Search ซึ่งเป็นโปรแกรมยูทิลิตี้โปรแกรมถัดไปนั่นเอง)

ผลจากการค้นหาของฟังก์ชัน strstr นั้น หากค้นหาพบก็จะส่งพอยน์เตอร์ที่ชี้ตำแหน่งของข้อความนั้นในพารามิเตอร์ตัวแรกมาให้ แต่หากค้นไม่พบก็จะส่งค่า NULL มาให้ ดังนั้นการทำงานของโปรแกรมหากการค้นหาข้อความ "/U" ในคอมมานด์ไลน์พบก็จะมีการกำหนดให้ตัวแปร Option เก็บตัวอักษร U และหากพบข้อความ "/L" พบก็จะให้ตัวแปร Option เก็บตัวอักษร L แทน ซึ่งตัวแปร Option นี้จะถูกใช้ในขั้นตอนของการเปลี่ยนตัวอักษร

จากตัวอย่างของโปรแกรมนี้ผู้อ่านสามารถใช้ฟังก์ชัน strstr ในการตรวจหาสวิตช์ข้อเลือกของการทำงานในโปรแกรมอื่นของผู้อ่านได้ต่อไป

ส่วนในการทำงานของฟังก์ชัน changecase( ) นั้น จะทำการอ่านไฟล์ข้อความมาครั้งละ 32 กิโลไบต์ โดยเก็บไว้ในตัวแปร buffer ซึ่งได้จองพื้นที่หน่วยความจำเอาไว้แล้วในฟังก์ชันเมน จากนั้นก็จะทำการเปลี่ยนตัวอักษรในตัวแปร buffer ทีละตัวอักษร โดยใช้ฟังก์ชัน toupper( ) หรือ tolower แล้วสวิตช์ที่กำหนดเลือกในคอมมานด์ไลน์ หลังจากเปลี่ยนตัวอักษรจนครบทั้งบัฟเฟอร์แล้วก็จะเขียนข้อความในตัวแปรบัฟเฟอร์คืนลงไฟล์



โปรแกรม TS (Text-Search)

การทำงาน

โปรแกรม ts จะช่วยในการค้นหาตำแหน่งบรรทัดของข้อความที่ต้องการในไฟล์ข้อความใด ๆ ซึ่งไฟล์ข้อความนี้สามารถระบุเป็นกลุ่มของไฟล์ได้ด้วย การค้นหาข้อความในไฟล์จะมีประโยชน์มากซึ่งนักเขียนโปรแกรมหลายท่านอาจเคยใช้งานโปรแกรม grep มาก่อน

หลายครั้งที่นักเขียนโปรแกรมต้องการทราบว่าโปรแกรมย่อยหรือตัวแปรต้องการค้นหาอยู่ในไฟล์โปรแกรมต้นฉบับใด และอยู่ในบรรทัดที่เท่าไร เพื่อจะได้เข้าไปแก้ไขหรือเปลี่ยนในไฟล์ที่ถูกต้องได้รวดเร็ว ซึ่งโปรแกรม ts นี้สามารถช่วยงานได้เป็นอย่างดี

การใช้งาน

```
C:\ >ts filename.ext tofind
```

filename.ext คือไฟล์ข้อความที่ต้องการจะค้นหา ซึ่งผู้ใช้สามารถระบุเป็นกลุ่มไฟล์ได้โดยใช้อักษรพิเศษ (Wildcard) ได้ด้วย ส่วน tofind คือข้อความที่จะใช้ค้นหา โดยข้อความดังกล่าวห้ามมีช่องว่าง (space) หากการค้นหานั้นพบ โปรแกรมก็จะพิมพ์ชื่อไฟล์ หมายเลขบรรทัดที่พบข้อความพร้อมทั้งข้อความในบรรทัดดังกล่าวออกมาให้

ตัวอย่าง

```
C:\ >ts *.c finddirs
```

เป็นตัวอย่างการค้นหาข้อความ finddirs ในกลุ่มไฟล์ภาษาซี C ทั้งหมด

```
A:\ >ts *.prg APPEND
```

เป็นตัวอย่างการค้นหาข้อความ APPEND ในกลุ่มไฟล์โปรแกรมดีเบส .prg ซึ่งหากโปรแกรมค้นหาข้อความในกลุ่มไฟล์โปรแกรมไม่พบ โปรแกรม ts จะทำงานเพียงพิมพ์ชื่อโปรแกรมเท่านั้น

```
C:\ >ts
```

หากเรียกชื่อโปรแกรมอย่างเดียว โปรแกรมจะแสดงวิธีใช้โปรแกรมให้

เทคนิคโปรแกรม

เทคนิคของโปรแกรม ts เป็นเทคนิคที่เคยใช้มาแล้วในโปรแกรมยูทิลิตี้ที่ผ่านมากหลายโปรแกรม เช่น การใช้งานฟังก์ชัน findfirst( ) และ findnext( ) การอ่านข้อมูลจากไฟล์ทีละ 32 กิโลไบต์ โดยใช้ฟังก์ชัน setvbuf( ) และอื่น ๆ อีก โดยเฉพาะกับการใช้งานฟังก์ชัน strstr ซึ่งได้อธิบายการใช้งานมาแล้วในโปรแกรม fc

ในฟังก์ชัน tosearch( ) จะมีการอ่านข้อความจากไฟล์มาทีละบรรทัด (ในโปรแกรมกำหนดความกว้างของแต่ละบรรทัดไว้ 80 ตัวอักษร ซึ่งหากไม่เพียงพอผู้อ่านสามารถปรับเปลี่ยนให้มากขึ้นได้) แล้วใช้ฟังก์ชัน strstr( ) เปรียบเทียบหรือค้นหาข้อความที่กำหนดจากคอมมานด์ไลน์ ซึ่งหากค้นพบโปรแกรมก็จะพิมพ์ชื่อไฟล์ หมายเลขบรรทัด และข้อความในบรรทัดนั้นออกมา แล้วค้นหาต่อจนกระทั่งหมดไฟล์

สำหรับฉบับนี้มุมภาษาซีขอเสนอโปรแกรมยูทิลิตี้เพียง 2 โปรแกรมก่อน หากผู้อ่านท่านใดต้องการไฟล์โปรแกรมต้นฉบับก็สามารถหาได้จากซอฟต์เมล์เช่นเดิม

รูปที่ 1 โปรแกรม FC

```
1 /*
2   Title          :  FC.EXE-File Case
3   PurPose        :  Will allow the quick conversion to either upper or lower case of an entrie file.
4                     Any text file will work.
5
6
7   Requirements   :  The program will computer using version 2.0 or work on MS-DOS and PC-DOS
8                     computer using version 2.0 or later
9
10  /*

12 #include <stdio.h>
13 #include <string.h>
14 #include <alloc.h>
15 #include <ctype.h>
16 #include <dir.h>
17 #include <io.h>
18
19 /*
20 Only internal procedure used in this program.
21 It performs the work.
22 */
23
24 void changecase (void);
25
26 /*
27 Global variables used throught this program. The variable 28  option will usually never get filled, but the size
is set 29  at 128 just to insure that there are no problems
```

→

```
 30 */
 31
 32 FILE     *tochange;
 33 int      handle;
 34 long     size,  numread, filepos;
                    35 char     filestr [MAXPATH], options [128], option [2];
 36 unsigned  char *buffer;
 37 char     *workstr;
 38
 39 /*
 40 Unlike the main loop in other previous utility program, this mains 41  performs quite a bit of the work. Besides printing a
copyright 42  message and allocating memory for a buffer, it also checks wether 43  the allocation was a success, wether there are
enough arguments on 44  the command line and parses the option on the command line for 45  valid option switch characters.
 46
 47 CALLED FROM: (DOS)
 48
 49 CALLS      : changecase
 50
 51 */
 52
 53 main (int argc, char *argv[])
 54 {
 55   int  j;
 56
 57             printf ("\nFC-File Case, TC Edition.\n");
 58
 59             buffer = (unsigned char *) calloc (32767, 1);
 60             if (buffer == NULL)
 61                   {
 62                   printf ("\nNot enough memory to run this utility. Get rig of"
      );
 63                   printf ("\nmemory resident programs and try again.\n");
 64                   return (1);
 65                   }
 66
 67             if (argc > 2)
 68                   strcpy (filestr,argv [1]);
 69             else
 70                   {
 71                   printf ("\nUsage:C:\>fc filename.ext [/u] [/l]\n");
 72                   printf ("\n        /u - for upper");
 73                   printf ("\n        /l - for lower");
 74                   return (2);
 75                   }
 76
 77             for (j = 2; j <= (argc - 1); j++)
 78                   strcat (options, argv[j]);
 79
 80             for (j = 0; j < strlen (options); j++)
 81                   options [j] = toupper (options [j]);
 82
 83             workstr = strstr (options, "/U");     /* Search for a u */
 84
 85             if (workstr)
 86                   {
 87                   option [0] = 'U';
 88                   option [1] =  0 ;
 89                   }
 90
 91             workstr = strstr (options, "/L");     /* Search for a l */
 92             if (workstr)
 93                   {
 94                   option [0] = 'L';
 95                   option [1] =  0 ;
 96                   }
 97             tochange = fopen (filestr, "r+");     /* Open file for update *
      /
 98
 99             if (tochange == NULL)
100                   {
101                   printf ("\nThere was a problem with the file. Please\n");
102                   printf ("check the spelling of the filename. \n");
103                   return (3);
104                   }                                                    →
```

รูปที่ 1
โปรแกรม FC (ต่อ)

```
105
106             handle = fileno (tochange);
107
108             changecase ();
109
110             fclose (tochange);
111             free (buffer);
112             return (0);
113 } /* main ends */
114
115 /*
116 Function changecase reads in consecutive chunks of the specified 117  file and converts the text in the file to either upper or
lower 118  case depending on the value stored in the OPTION.
119
120 CALLED FORM:   main
121
122 CALLS    :  (none)
123 */
124
125 void changecase (void)
126 {
127             unsigned int i;
128             int     numtoget;
129
130             size = filelength (handle);
131             numread = 0;
132
133             if (size <= 32767)
134                     numtoget = size;
135             else
136                     numtoget = 32767;
137
138             do
139                     {
140                     filepos = ftell (tochange);
141                     fseek (tochange, numread, 0);
142
143                     _read (handle, buffer, numtoget);
144
145                     for (i = 0; i <= numtoget; i++)
146                       {
147                       if (option [0] == 'U')
148                         buffer [i] = toupper (buffer [i]);
149                       if (option [0] == 'L')
150                         buffer [i] = tolower (buffer [i]);
151                       }
152
153                     fseek (tochange, filepos, 0);
154                     _write (handle, buffer, numtoget);
155
156                     numread += numtoget;
157                     if ((numread + numtoget) > size)
158                       numtoget = size - numread;
159                     }
160             while (numtoget != 0);
161 }
```

รูปที่ 1
โปรแกรม FC (ต่อ)

```
Title          : TS.EXE - Text search
Purpose        : This utility will search for a text string
                 which you specify in he files you want.
                 You can specify the file or use wildcaed
                 characters for a search through a group of
                 related files.
Requirements   : The program will work on MS-DOS and PC-DOS
                 computer using version 2.0 or later
```

→

รูปที่ 2
โปรแกรม TS

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>
#include <dir.h>


/*
Only internal procedure used in this program.
*/

void search (void);

/*
All variable are declared global.
*/

FILE        *tosearch;
char        searchstr [MAXPATH], losttext [128],
            pathname [MAXPATH],  drive [MAXDRIVE], subdir
            [MAXDIR],    file [MAXFILE], ext [MAXEXT],      name
            [MAXPATH];

            char                 *inbuffer;
            int                  done;

            struct       ffblk dta;

            /*
This main loop sets up the buffer, performs extensive error checking, opens the file and call the process function  to search the file
for a text string. Only text files will work properly.


CALLED FROM:  (DOS)

CALLS              : process
*/

main (int argc, char *argv[])
{
      printf ("\nTS-Text Search, TC Edition. \n");

      if (argc > 2)
            {
            strcpy (searchstr, argv [1]);
            strcpy (losttext, argv [2]);
            fnsplit (searchstr, drive, subdir, file, ext); sprintf (pathname, "%s%s", drive,
            subdir);
            }
      else
            {
            printf ("\nUsage:C:\>ts filename.ext tofind\n"); printf ("\n        tofind  - searching string\n");
            return (1);
            }

      inbuffer = (char *) calloc (1, 32676);

      if (inbuffer == NULL)
            {
            printf ("\nError - problem allocating space for buffer.\n\n"); return (2);
            }

      done = findfirst (searchstr, &dta, 47);

      while (!done)
            {
            strcpy (name, pathname);
            strcat (name, dta.ff_name);
            strupr (name);
```

รูปที่ 2
โปรแกรม TS (ต่อ)

```
            tosearch = fopen (name, "r");

            if (tosearch == NULL)
                  {
                                    printf ("\nError - problem opening the file.\n\a");
                  return (3);
                  }

                                    if (setvbuf (tosearch, inbuffer, _IOFBF, 32676) != 0)
                  {
                  printf ("\nError - problem setting up input buffer.\n\a"); return (4);
                  }

            search ();

            fclose (tosearch);

            done = findnext (&dta);
            }

      free (inbuffer);

      return (0);
} /* main ends */

/*
This function processes the line from the input file. It will search every line read for the charater string that the user specified
at the command line.

CALLED FROM: main

CALLS             : (none)
*/

void search (void)
{
      /*
      two variables specific to this function
      */

      char line [81];
      int  i, ch;

      i = 0;
      while (fgets (line,80, tosearch) != 0)
            {
            i++;
            if (strstr (line, losttext) != NULL)
                  {
                  printf ("\nfile: %s\t line number: %d\n%s\n", name, i, line); printf ("Any key to continue or ESC to quit.\n");
                  ch = getch ();

                  if (ch == 27)
                        {
                        fclose (tosearch);
                        free (inbuffer);
                        exit (0);
                        }
                  }
            }
}
```

รูปที่ 2
โปรแกรม TS (ต่อ)